

**Version 1.1**

**Date: 2018/12/09**

Abstract

**The objective of this document is to describe the Catalogue Maintenance application that supports structures and applications driving the sales decision support business requirements**

Catalogue Maintenance

*Define All Structures and Application supporting lookup demands*

**Table of Contents**

Document approval and distribution list 2

1. Introduction 3

2. Audience 4

3. Objectives 5

4. Database Design Objectives 5

5. Data normalisation 8

6. Multiple supplier for same part 9

7. Alternates, supersessions and replacement parts 9

8. Dependencies 10

9. Design philosophy 10

10. Database design philosophy 10

11. The advanced searching approaches 11

12. Catalogue lookup to sales-order 12

13. Database entities and relationships 14

14. Programs 16

14.1 MS Windows Executables 16

14.2 SQL Stored Procedures 16

15. Acceptance 17

# Document approval and distribution list

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Name / Title** | **Signature** | **Date** |
| **Document Type / purpose** | | | |
| Prepared by |  |  |  |
| Reviewed by |  |  |  |
| Approved by |  |  |  |

# Introduction

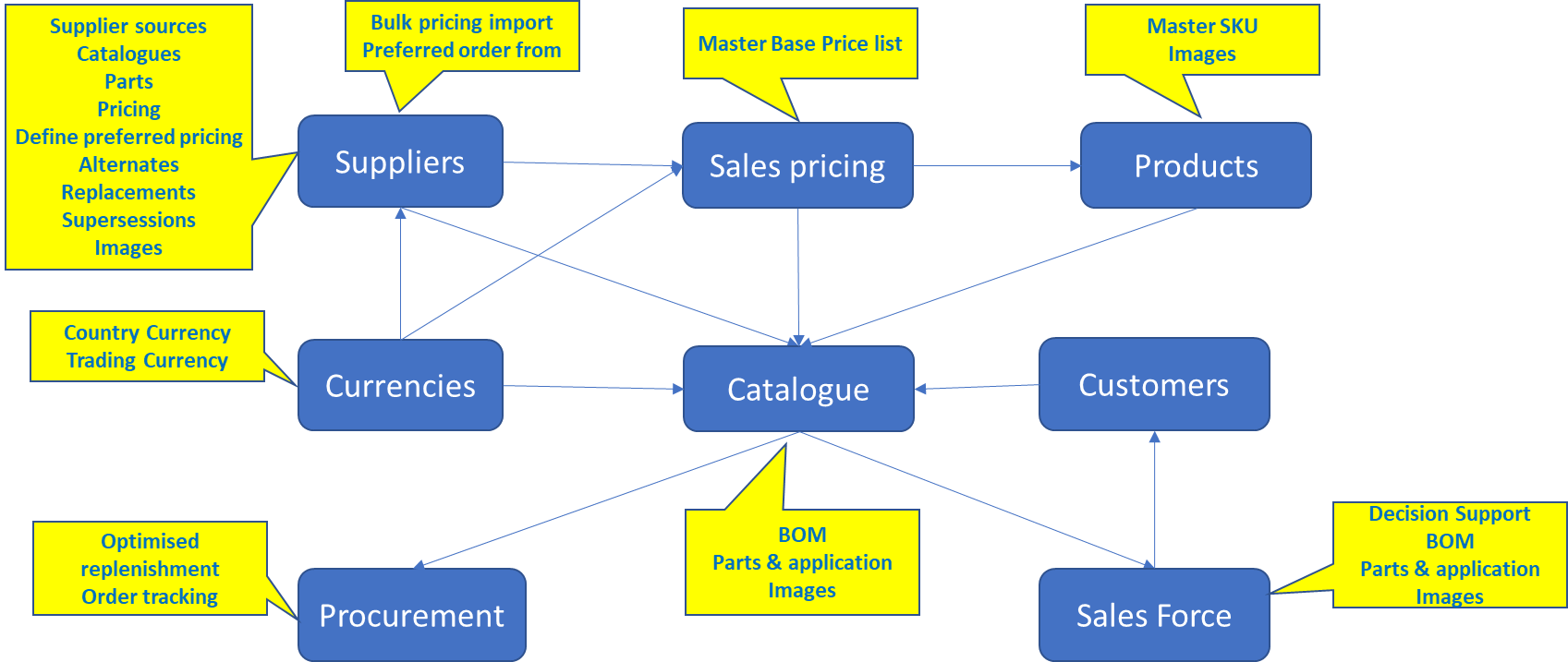
The so-called **Catalogue Lookup**represents an end user tool that is able to promote sales of parts to the industry through systems-based intelligence that is created using data captured in various sub systems in ePart.

Systems functionality exists to consolidate the data in contained in the various subsystem in an optimised manner in support of business requirements.

Additionally, there is functionality to create assemblies and sub-assemblies that represent the construct of parts and their relevant applications.

The following diagram provides a high-level end to end view of the ePart Catalogue system with some indicative functionality guideline annotations relative to sub-system functionality.

**Diagram 1.1: *A high level view of the ePart catalogue sub-system integration***



There is no ***redundant data in ePart due to the end to end systems and application integration***, consequently when reviewing the ePart ***catalogue*** end to end, it may lead to a misconception that it is complex. However, by brining all the sub-system components together in a unified manner it exposes the inter sub-system dependencies properly and negates the need to re-capture the same data repeatedly.

About the ***catalogue*** maintenance section, there are a number of functions that ease the repetitive work of catalogue engineers; some of these are:

* Creating tree structures of assemblies and sub-assemblies in line with the manufacturing industry and resembles the motor manufacturing industry method
* Creating assembly templates that can be re-used within the assembly tree
* Inheritance of template assemblies onto parent assemblies with or without future change inheritance.
* Example of this is the Ford 1000cc engine used in more than one vehicle. The engine assembly is created as a template and then linked onto the various vehicles where it is used. After linking, the linked version can be changed for subtle variances specific to the linked to vehicle. Although the example is for vehicle and engine, the same principal can be used at any level within the tree structure.
* Assemblies may be SKU items that can be sold i.e. turbo charger. However, a turbocharger may have sub-assemblies that could be selling SKU’s as well i.e. shaft bearings. Thus, the tree allows for the opportunity to define sellable SKU items as well as the constituent components that make up the SKU assembly.
* It must be noted that one or more images can be added at SKU level and at assembly level in the tree. The assembly level image would be applicable where an image would assist the selling process for the question ‘Does the bearing fit this gearbox?’ and we do not stock gearboxes.

# Audience

Sales

Cataloguing

Management

# Objectives

To optimise the definition process of creating structures and relationships that depict the ***real-world*** of stocking items and their related applications. In support of this, the database structures are defined using an ***inverted tree structure to manage the so-called bill of material definition.***

Follow the real-world parts identification allowing these to resolve back to the ePart base number (SKU)

Provide parts and assembly images to assist in the identification process, both for purchasing and sales teams.

Include as much detail as is possible so that operationally the business is optimised through the availability of pertinent information at the time. Illustratively, tariff heading in support of imported goods as well as for the export to neighbouring countries.

Have relevant dashboards through which the underlying data can be observed, and management decision made to improve business processes i.e:

* Selecting best supplier option for replenishment
* Define which supplier pricing to be used when calculating standard selling price
* Define which supplier pricing to be used when calculating wholesale price lists
* View sales trends to optimise pricing overall
* ….

# Database Design Objectives

Analysis of the data available, data gathering from various sources, relationships of parts to applications, it became evident that a ***formalised hierarchical structure*** (inverted tree) would be important in the data persistence implementation to support operational functionality.

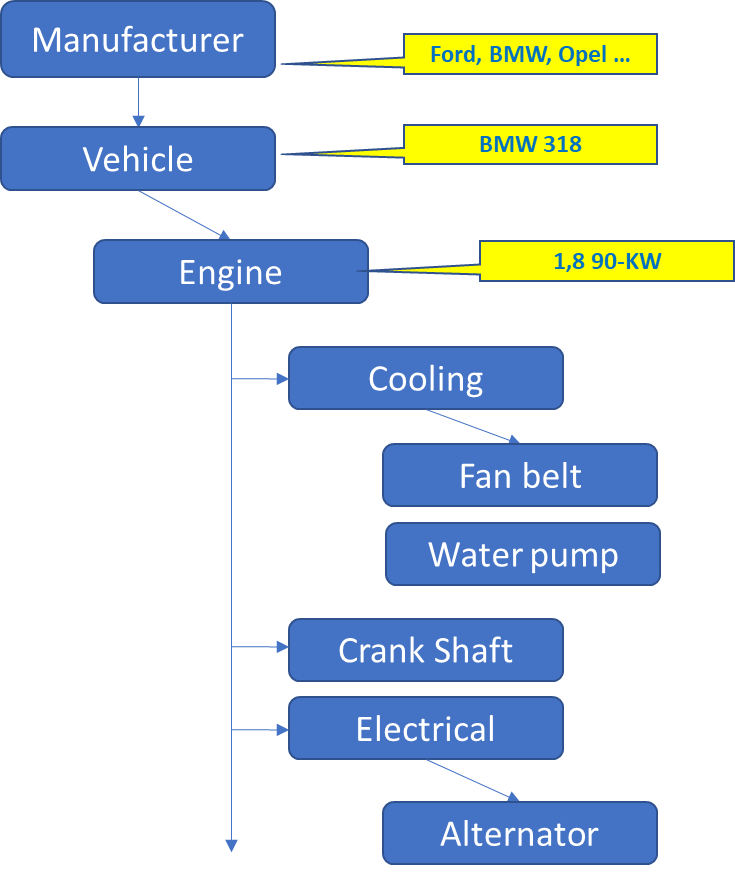
Contained in the legacy (COBOL based) catalogue that was to be replaced, was details of some part to application details. Although simplistic in representation, there was enough textual detail in the parts description to infer an inverted tree representation which was thought to be a suitable fit for the data available.

Some of the historical experience in the developer team was to develop solutions with open ended parent child relationships which does solve the visual view of the parts to application relationships not unlike a computer system inverted tree of directories and sub-directories i.e. Windows.

The following image provides a high-level view of how it would apply in the ePart catalogue system.

Inherently, the design is the inverted tree system is totally agnostic in terms of the specific assemblies and related sub-assemblies and parts linked. The implementation is quite capable of defining a food menu (pizza anyone) and link the recipe and ingredients to make a specific pizza on a menu!

This then allows the organisation the opportunity to create many bills of material representative of specific applications and their underlying catalogue parts.

**Diagram 4.1: *Provides a high-level view of the hierarchical data representation needed for parts to application relationships***

The following simplified diagram illustrates the database implementation that supports the inverted tree BOM implementation with linking to parts (SKU) as part of the leaf nodes:

**Diagram 4.2: *Inverted tree with leaf node parts linking***

This ***self-referencing*** relationship provides for unrestricted levels of relationships to be assembled in an ***assembly*** to ***Sub-Assembly*** without limits. The ***Part(s)*** relationship provides the ***item level component stocking attributes*** as depicted in the following diagram:

**Diagram 4.3 provides a view of how the relationship is used to depict an easily viewed construct**

The assembly / sub-assembly explosion indicates how elements with related attributes can be viewed with the highlighted entry indicating the actual stock item.

By design the solution provides for the linking of assemblies to stocking items even when the stock item is made up of sub-assemblies i.e. a turbo charger can be sold as a complete unit yet some parts making up the turbo charger can be sold separately.

To add a level of governance, there are a number of system and application defined rules to prevent end-users from making obvious mistakes i.e. linking a vehicle to an engine or linking a turbocharger to a turbocharger to turbo shaft bearing.

Additionally, the application make provision for the creation of ***template*** assemblies (usually done by senior staff with experience) that are then used by cataloguers through ***inheritance.*** This means that when declaring a new vehicle application, the work does not need to be repeated – use the appropriate template.

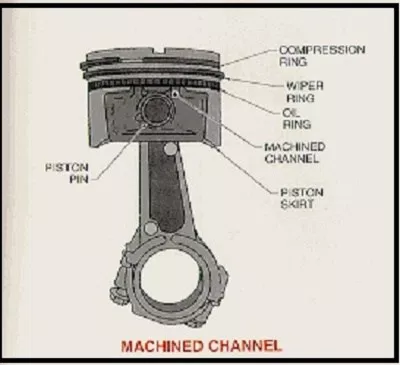
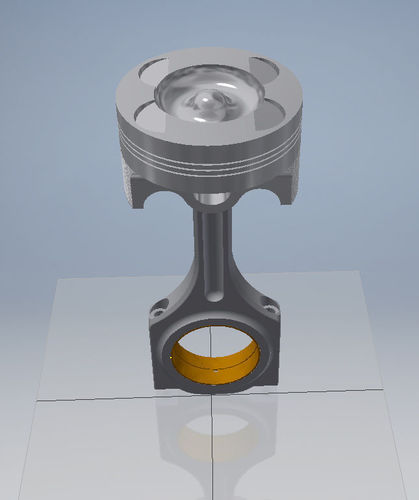
***Note that there are more control attributes such as assembly type, code and others. Refer to the full entity relationship chart for more details***

# Data normalisation

When the database design was undertaken, the understanding was gained that the traditional data normalisation (5th normalisation) against each product type could not succeed against the variety of SKU types in existence. This in support when searching for a suitable bearing size vs a piston of a type.

By way of an example, a piston as a type has properties that are generally repeatable between pistons of different piston applications i.e. attributes types of the following:

* Diameter
* Skirt length
* Compression rings
* Wiper rings
* Oil rings
* Gudgeon pin size
* ….

On the other hand, ball bearings also have repeatable attributes between ball bearing types with attributes as follows that do not match those of a piston:

Speed Ratings

Reference speed: 75000 r/min

Limiting speed: 48000 r/min

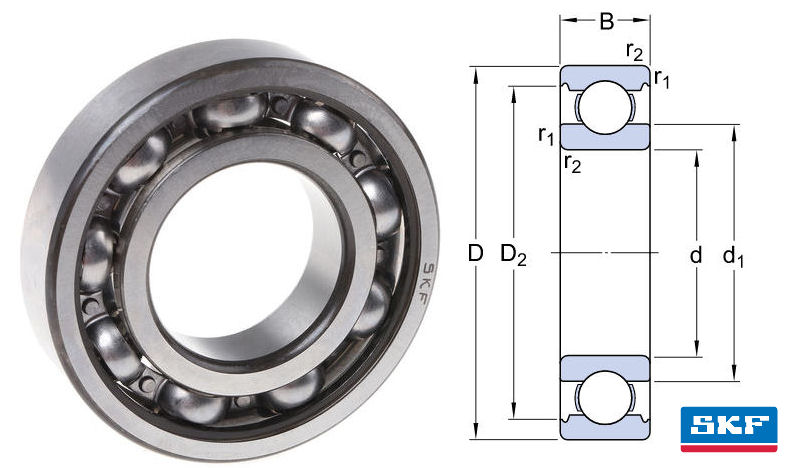
Dimensions mm

d1: 12.1

D1: 17.6

D2: 19.2

r1,2 min: 0.3



From a technical perspective the vast number of SKU (base numbers / parts) make a standard data normalisation approach an almost impossible task.

The design team of the day opted to allow the SKU properties to be user definable where a SKU is assigned a type and one or more properties against the type defined. Consequently, the data capturing program is SKU type aware and will limit the user input to the defined properties. This removed the need to perform data normalisation on a formalised manner but retained the ability to perform lookup searching against the defined attributes as well

# Multiple suppliers for same part

As part of the catalogue maintenance sub-system is to work through parts catalogues received from various manufacturers / suppliers. Details pertaining to the items listed in the catalogue are captured or updated without necessarily linking said part number to a system SKU (base number). It is assumed that at least that if there is a reference it will trigger a need to do further research to identify exactly the part and its application.

If at any time the un-linked part is identified against a SKU (base number) or another supplier part reference which is linked, the un-linked item is automatically cascade linked using the qualified detail exposed.

This feature allows purchasing staff to identify alternate suppliers from whom to order from.

It is a reality that OEM allocate part identifications per their Design & Manufacturing process that is not used by replacement part manufacturers – they use their own part identification system. However, the alternate part manufacturers do provide OEM referencing.

In support of this, ePart catalogue maintenance allows the creation of alternate identification part numbers from various suppliers / manufacturers. These linked back to the ePart Base number (SKU). Should a query be launched using the catalogued industry part reference, the ePart Base number can be resolved and hopefully the business concluded with success.

In the case of non-local suppliers (import), enough detail can be captured to allow for the export documentation to be completed i.e. tariff headings amongst others. Additionally, the product industry application detail (agricultural, commercial etc) needs to be maintained for government purposes so that import duties can be accounted for.

As part of the captured data, supplier parts profile is catered for i.e. dims, weight etc. Also, and of importance is the opportunity to capture the container bar code reference assigned to the supplier part. This bar code is linked to the SKU (Base Number) and can result in eliminating the print requirement for the ePart barcode.

Supplier bar codes can be captured as well allowing for speedier receiving

(Note: a Scoping document has been produced and needs management approval to execute on)

# Alternates, supersessions and replacement parts

As the assigned engineering skilled staff review their catalogue data, provision is made to mark parts in the following manner:

* Alternates is where a part is not the same but is otherwise a good alternate to use i.e. fan belts may have differing lengths but can do the same work.
* Supersessions is where a part is superseded for some reason by a newer part, it may very well be that the both the superseded and supersession are still serviceable. However, in the ePart system there is a specific flag setting that indicates that the superseded part may not be sold and will stop a sales transaction.

# Catalogue source material

Due to the large volume of source material received from manufacturers and distributors worldwide, it was a complex task to manage the material processed and not processed.

To ensure proper tracking, the catalogue source material is classified as if it were a public library. Each item (book or CD / DVD) was logged with various properties such as source of material, date received, number of pages etc into the ePart system.

Additionally, much of the material of late resides on the internet where paper based and to an extent CD/DVD are much reduced. To cater for these, the current system still applies in that the Web URL is captured with related properties.

Various dashboards / reports are available to reflect the status of each item in terms of date received and related progress

When reflecting of the ***lost sales*** due to part references not in record, the system is able to provide this kind of statistic and a manual process followed to allow the catalogue management to focus on these kinds of lost sales indicators.

# Catalogue maintenance work flow

To ensure that catalogue processing of source material moves forward against a given expectation, pages processed against the library total pages provides management with such statistics.

Paper based catalogues, based on management selection, are sent to printing companies for guillotining where the spine of the catalogues is cut releasing the catalogue pages. These loose pages are then passed through a high-speed duplex scanner for processing.

The scanned images are related to the reference data in the catalogue library and stored referentially. This means that the images are not stored in the database but externally with the database containing the location where the images can be found.

***Developer note***: If the images were stored in the database this would have a significant increase in the database size with no operational benefit. In fact, it would detract from the time to backup and especially time to recover in case of a database failure.

Each catalogue team member ***books*** an item out of the library and records each page processed. In the event where a paper-based book is selected, the scanned images are used directly off the computer screen.

At appropriate times, the cataloguer can select images on a specific page being processed by identifying the area and location of the image. Again the database does not contain the image at all; only the catalogue reference, page number, image co-ordinates and size is recorded in the database. Later when the image is requested, the page is retrieved and the image ***cut*** from the page image using the stores reference, co-ordinates and size. The selected are is the only part transmitted through the network. It is a proven benefit given the wide area network through which our customers operate from.

Pages are processed using text-based capturing (always) and where needed also adds product images as per the catalogue.

In the case of CD/DVD, these are pre-processed on a priority base as defined by management. Each ***PAGE*** is selected and recorded on ePart. This is a lot more time consuming than paper-based catalogues due to the inability to process the CD/DVD data.

For the internet-based catalogues the same scenario exists where the page based extraction does not exsist.

Considerable effort was made to get suppliers to drop to ePart files that are page based as if scanned without ANY success.

A further investment was made to try and extract the individual pages using PDF reader software with limited success.

Similarly, the web-based extraction has yet to succeed. However, it would be good governance to see if better tools may be available to extract web-pages better. Some investment towards this is a business requirement.

# Supplier Pricing

Generally, supplier pricing is received in bulk format with periodic adjustment post receipt of bulk pricing.

For the bulk pricing there is a efficient process of pricing updaters importation with related error flagging where exceptions may be found. Generally, exceptions are around the current pricing model in ePart that calculate to an invalid sales price basis, usually under cost or too small a margin.

Similarly, the occasional price updates can be captured through a user interface with appropriate error reporting.

Post supplier price update, the active suppliers and related pricing is reviewed, and decisions made as to which supplier may be the flagged preferred supplier. Also, there is a flag that is set to define which supplier will be the preferred selling price basis to calculate from, this includes the defining of the ePart wholesale selling price.

To note, the pricing is always expressed in ZAR from the sales perspective even though the suppliers may be using their own currency for creditors settlement.

# Currency changes

Due to the fact the Engineparts has the ability to order stocking items internationally, up to date currency conversion factors are a reality.

The ePart system has been optimised to apply changed conversion factors on the fly and is able to apply such updates in less than 2 seconds whereas in the past (pew ePart) this would have consumed several weeks of preparation and processing to recalculate the selling pricing.

Provision has been made but not implemented, to allow standard costing apportionment to GL for cost incurred beyond the currency factors for attributes such as forward cover, insurance etc.

# Base cost calculations – optimisation

# Recall of faulty / safety risk goods sold and on hand

# Supplier part profiles

# Dependencies

|  |  |  |
| --- | --- | --- |
| # | Description | Action / By whom |
| 1 | Accounts receivable |  |
| 2 | Pricing |  |
| 3 | Catalogue maintenance |  |
| 4 | Discount structures |  |
| 5 | Warehouse activities |  |
| 6 | Sales order |  |

# Design philosophy

The design philosophy closely adheres to the manner in which the ePart was incarnated by splitting the entire application into 3 basic components:

* 1. Presentation – this is done in Builder c++ with limited if any engagement of business logic
  2. Business logic – this is done using MSSQL stored procedures
  3. Data persistence – the fact that business logic is contained in Stored Procedures makes the persistence integral part of the business logic process.

The underlying benefit to this approach is simplicity and maintainability.

The obvious notion to this is that the presentation logic can theoretically be swapped out retaining the business and persistence logic intact.

# Database design philosophy

In the after-market automotive industry the concepts of assemblies, sub-assemblies and finite parts are strongly represented.

Consequently, the ePART catalogue contains all parts in relation to a tree of assemblies, called the bill of materials, or BoM.

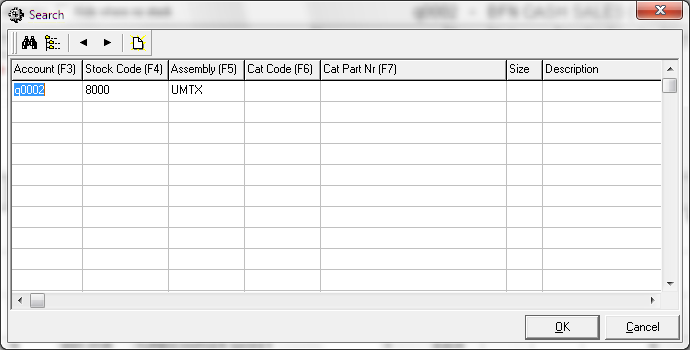
**Diagram 8.1 provides a high-level view of the *core* catalogue Entity Relationship as deployed**

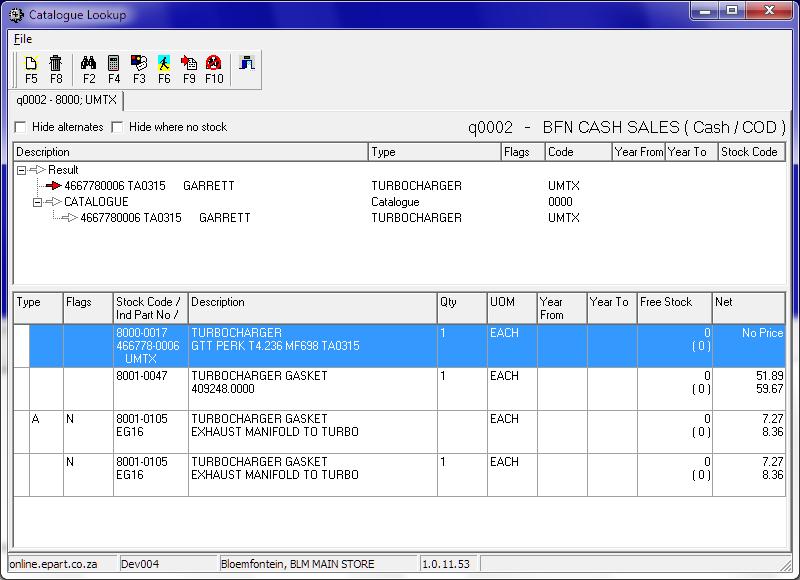


# The advanced searching approaches

In addition to the parts explosion (inverted tree) format of searching, there is the advanced searching capability and is depicted in the following diagram:

**Diagram 9.1: A high-level view of advanced search capabilities:**



**Diagram 9.2: The search results:**

Some of the search criteria can expose the concept of ***KITS*** being for servicing or engine overhaul and several others as well. This allows for opportunistic selling by the sales staff enhancing the customer experience.

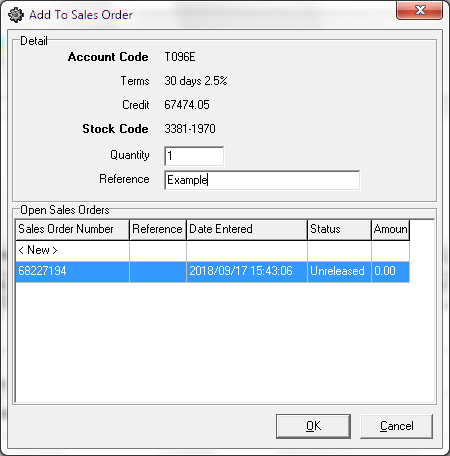
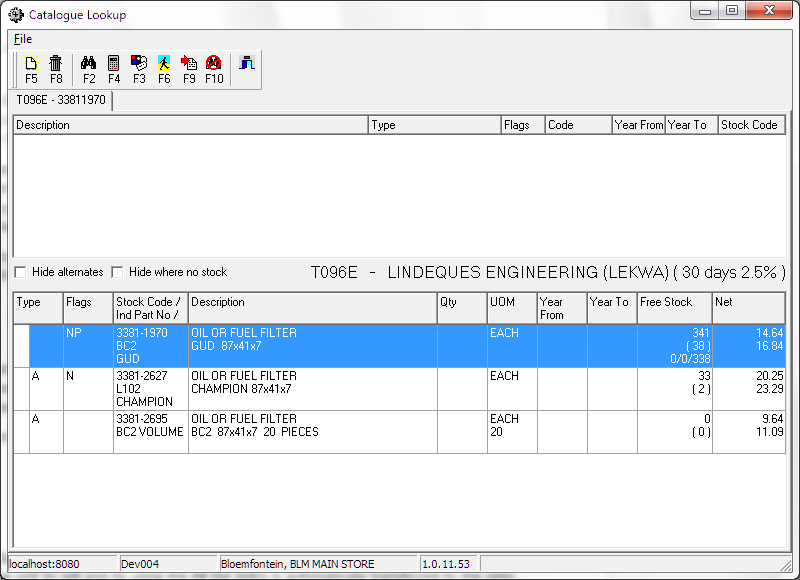
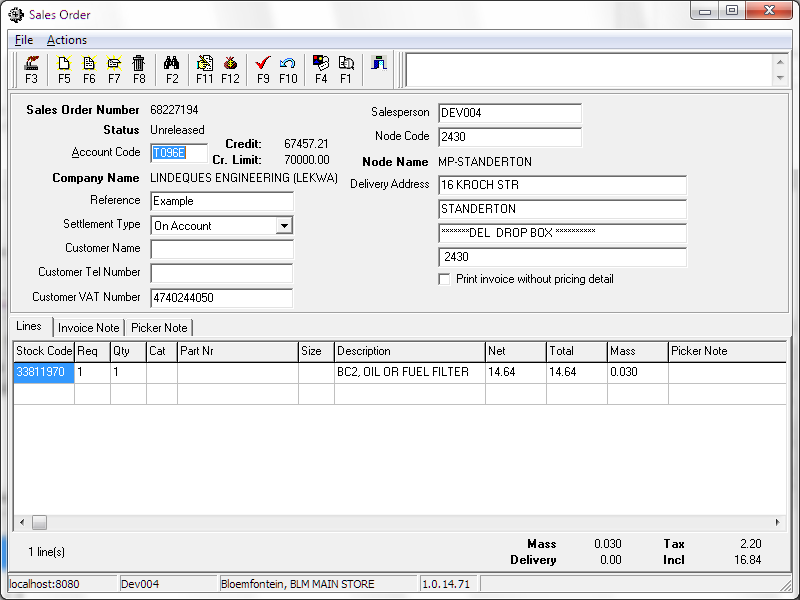
The catalogue lookup allows for the searching using industry part numbers or supplier part numbers for the same stocking unit. This is for instance a piston ring can have multiple industry part numbers depending on which manufacturer distributer is called off by the customer.

To enhance customer experience, the system allows for supersessions, alternates and discontinuations.

# Catalogue lookup to sales-order

The catalogue lookup application operates independently from the sales order program. However, there is a path of communication from the catalogue lookup to the sales order as depicted in the following diagram

**Diagram 10.2: A view of how the lookup interacts with the sales order**



The effective optimisation is that from the search and filter results set, a choice is made of the stocking unit to sell and by using the ***F9*** the entry is automatically transferred to the sales order.

# Database entities and relationships

The full catalogue related database diagram provides a view of the various participating relationships. Notably is the simplicity of the database tables participating in the core structure.

Many of the additional tables are there for control and optimisation purposes.

A specific reference is made to the following tables:

* “Bridging table” – this is often used in data mining structures where the volume of data requires a specialised table of this kind
* BOM level – this table is a control table to prevent illogical parent child linking i.e. linking an engine to a water-pump rather than linking a water-pump to an engine
* ….
* ….
* ….
* …

The catalogue support system provides for a highly efficient source of information, provided the underlying data is accurately maintained and kept up to date; failing which will result in poor customer experience.

# Programs

# MS Windows Executables

|  |  |
| --- | --- |
| **Name** | **Description** |
| catLookup2.exe | Used to look up which parts the customer wants to purchase, and add them to the sales order. |

# SQL Stored Procedures

|  |  |
| --- | --- |
| **Name** | **Description** |
| catCatQueryResultHomeFindByQueryNo2 | Searches for items matching the criteria, to populate the item grid. |
| catCatQueryBomNodeHomeFindByQueryNo | Finds the section of the BoM above and below the search results. |
| salAddToSalesOrder | Automatically adds an item to an existing or new sales order. |
| catLogLostSaleManually | Records a formal lost sale, in the form of a failed-search-type informal lost sales marked as “manual”. |

# Acceptance

I hereby confirm that I have been fully informed of the document’s content and received training to understand how the detailed instructions are to be applied:

Name …………………………………………………………………………….

Job Title ………………………………………………………………………….

Signed ……………………………………………………………………………

Date ………………………………………………………………………………